DOCUMENT RESUME

ED 260 906 SE 045 943

AUTHOR Richardson, Michael; Hunt, Earl
TITLE Problem Solving Under Time-Constraints.
INSTITUTION Washington Univ., Seattle. Dept. of Psychology.
SPONS AGENCY Office of Naval Research, Arlington, Va. Personnel and Training Research Programs Office.
REPORT NO ONR-TR-10
PUB DATE Aug 85
CONTRACT N00014-84-K-5553
NOTE 58p.; Paper presented at a Meeting of the Mathematical Psychology (San Diego, CA, August, 1985). Document contains several pages of light type.
PUB TYPE Reports - Research/Technical (143)

EDRS PRICE MF01/PC03 Plus Postage.
DESCRIPTORS *Cognitive Processes; College Students; *Computer Simulation; Educational Research; *Mathematics Instruction; *Metacognition; *Modeling (Psychology); *Problem Solving; Time Factors (Learning)
IDENTIFIERS *Mathematics Education Research; Mental Computation

ABSTRACT

A model of how automated and controlled processing can be mixed in computer simulations of problem solving is proposed. It is based on previous work by Hunt and Lansman (1983), who developed a model of problem solving that could reproduce the data obtained with several attention and performance paradigms, extending production-system notation to time-constrained problem solving. Four college students were required to do mental arithmetic rapidly, keeping track of the running total of a visually presented series of numbers using base three (to keep the task from being fully automated). From time to time, they had to deal with an interruption and then return to the arithmetic task. The students generally, but not always, managed to remember the things they needed to remember, and often became aware of an error after making it, indicating there was a good deal of self-monitoring of behavior. The problem solver's internal control of information seems to be an essential part of real-time problem solving. These processes were reflected in the simulation model, which is described in detail. (MNS)

****
* Reproductions supplied by EDRS are the best that can be made *
* from the original document. *
****

U.S. DEPARTMENT OF EDUCATION
NATIONAL INSTITUTE OF EDUCATION
EDUCATIONAL RESOURCES INFORMATION
CENTER (ERIC)

☑ This document has been reproduced as received from the person or organization originating it.

☐ Minor changes have been made to improve reproduction quality.

• Points of view or opinions stated in this document do not necessarily represent official NIE position or policy.

"PERMISSION TO REPRODUCE THIS MATERIAL HAS BEEN GRANTED BY

Office of Naval Research

TO THE EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)."

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: Technical Report 10
2. GOVT ACCESSION NO.:
3. RECIPIENT'S CATALOG NUMBER:
4. TITLE (and Subtitle): Problem Solving Under Time-Constraints
5. TYPE OF REPORT & PERIOD COVERED:
6. PERFORMING ORG. REPORT NUMBER:
7. AUTHOR(s): Michael Richardson and Earl Hunt
8. CONTRACT OR GRANT NUMBER(s): N00014-84-K-5553
9. PERFORMING ORGANIZATION NAME AND ADDRESS: Department of Psychology, University of Washington, Seattle, Washington 98195
10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS:
11. CONTROLLING OFFICE NAME AND ADDRESS: Personnel and Training Research Programs, Office of Naval Research, Arlington, Virginia 22217
12. REPORT DATE: August 15, 1985
13. NUMBER OF PAGES:
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office):
15. SECURITY CLASS. (of this report): Unclassifed
15a. DECLASSIFICATION/DOWNGRADING SCHEDULE:
16. DISTRIBUTION STATEMENT (of this Report): Approved for public release; distribution unlimited.
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report):
18. SUPPLEMENTARY NOTES: Paper presented at the Mathematical Psychology meeting, August , 1985, University of California, San Diego.
19. KEY WORDS (Continue on reverse side if necessary and identify by block number): Thinking, problem solving, dual tasks, computer simulation, production systems, arithmetic.
20. ABSTRACT (Continue on reverse side if necessary and identify by block number):

The computer science concept of production execution has been the basis of a large number of simulations of human problem solving. Typically these simulations have operated in a timeless environment, in the sense that they did not consider any constraints to solve problems quickly. In a previous technical report in this series Hunt and Lansman proposed an architecture for production-executing machines that could be applied to real-time problem solving. Hunt and Lansman supported their ideas by simulating data from laboratory studies

(over)

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE S/N 0102-LF-014-6601

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

drawn from the attention and performance field. In this paper Hunt and Lansman's approach is extended to the simulation of people doing a simple arithmetic task under considerable time pressure and when subject to interruptions. Data from the simulation is compared to data from college students doing the same task.

PROBLEM SOLVING UNDER TIME-CONSTRAINTS

Michael Richardson

and

Earl Hunt

University of Washington

INTRODUCTION

Problem solving sometimes takes place under severe real-time constraints. Consider the problem of an automobile driver approaching an intersection. If the light is either red or green a highly overlearned response is appropriate. Little in the way of problem solving is required. If the light is yellow a decision must be made. The decision is based on several variables; the distance to the intersection, the speed of the auto, the speed and distance of any following auto, and perhaps its identity (e.g. is the following auto a police car?) A good bit of reasoning has to be completed, quickly, or the driver is

BEST C

likely to have a brief and unhappy career.

How is it that people perform such tasks quickly and, for the experienced, with very few errors? Saying that skilled performance is 'automatized' is unsatisfactory. To be sure, in many cases real-time performance is fast and accurate, a critical descriptive attribute of automated behavior (Schneider and Shiffrin, 1977.) However a crucial precondition, consistent mapping between stimulus and response, is often not met. Consider again the driving example. When people drive in heavy traffic without accidents, they continually face new and complex situations. Furthermore, it is clear both from intuition and experimental results (Brown and Poulton, 1961) that driving diverts attention from other ongoing activities. By definition, this means that the behavior is not automated.

The driving example is an illustration of a more general issue. Time-constrained problem solving depends upon a mixing of controlled and automatic processes. The mixing must take place rapidly enough to keep up with real-world demands on the problem solver. However, most computer simulations of problem solving do not consider the influence of real-time constraints on thought.

OPY AVAILABLE

"Reaction time" studies are more traditionally assigned to the attention and performance field. Which aspects of the processing are automated? What are the control mechanisms and under what conditions is control exercised?

Here we present a model of how automated and controlled processing can be mixed. The model is based on previous work by Hunt and Lansman (1983), who proposed a model of problem solving that could reproduce the data obtained with several attention and performance paradigms. Their model is an extension of the widely used production-system notation to time-constrained problem solving. A production-system model of a task can be divided into two distinct components. One is the set of productions that are used to specify actions in the task itself. These will be specific to a given situation. The other component is the set of mechanisms that control production execution. A good analogy is to regard the productions used in a simulation as a program to model task specific behavior and to regard the mechanisms of production execution as a model of the content-free information processing mechanisms that must underlie all thought. Following Pylyshyn (1984), we shall use the term "functional architecture" to refer to the information processing mechanisms collectively.

BEST

What Hunt and Lansman did was to develop a functional architecture for production execution that was based upon models that have been proposed to explain "attention and performance" phenomena, such as dichotic listening, Hick's law, and response choice in the face of conflicting cues. Hunt and Lansman then used this architecture to execute a number of simple production systems, corresponding to a participant's understanding of the instructions that might be given in a variety of attention and performance tasks. The time required to execute the production system program was produced by an interaction between the logic of the production system and the mechanisms of the functional architecture, which was constant across tasks. Hunt and Lansman showed that the interaction could mimic human data in fairly simple laboratory tasks, including choice reaction time paradigms, a divided attention task, and a simple version of the Stroop paradigm. They suggested that the same approach could be applied to understand real time problem solving in more complex situations, but they did not carry their work to the point of simulating the more complex situtations.

This paper reports an extension of Hunt and Lansman's model to a situation in which people must do mental

OPY AVAILABLE

arithmetic rapidly. In addition, the participants in our work had to deal with a simple interrupting task. More specifically, college students kept track of the running total of a series of numbers. To keep the task from being a fully automated one, we required that the total be kept in base three (trinary) arithmetic. From time to time the arithmetic task was interrupted. The participants had to deal with the interruption and then return to the arithmetic task. This situation was chosen as a compromise between the extremely complex situations, such as driving, that we would like to model and those tasks that can be managed in a laboratory situation.

THE TASKS AND THE PARTICIPANT'S BEHAVIOR

Procedure

Participants were first trained to do Base 3 (Trinary) arithmetic. They were then given the task of keeping a running total of a series of visually presented trinary numbers. From time to time during the presentation of the numbers to be added an auditory signal was presented. The participant was to press a button in response to the tone as soon as it was detected.

Three dependent variables were studied. These were the speed and accuracy of the arithmetic responses and the reaction to the tone.

Addition Task.

Subjects Four University of Washington undergraduate students were recruited through advertisement. They were paid for their time.

Training and instructions. Training included memorization of the basic addition facts (e.g. 1 + 2 = 10)

BEST COPY AVAILABLE

and sufficient practice with both written and verbal problems so that all participants achieved a perfect score on a 120 item written test. The problems involved one to four digit integers. Some of the problems required carries across one or more columns.

The participants were instructed to do the problems in trinary notation directly, rather than solving a problem in decimal notation and then converting the results to trinary notation. They were also instructed to do multiple digit problems from right to left, one column at a time. Both of these instructions were intended to prevent subjects from adopting idiosyncratic methods for particular addition problems. For example, during training some subjects reported recalling from previous trials the sum of 122 and 1 (i.e. 200) without intermediate carrying from column to column. All participants reported they had no difficulty in complying with the instructions.

Participants were also instructed to emphasize both speed and accuracy, that is, '...to respond as quickly as possible but try not to make mistakes.'

Experimental procedures. On each trial, the stimulus

BES

digit (approximately 2.1 mm x 4 mm) was presented on a video monitor located directly in front of the participant, at a distance of approximately 75cm. The persons's task was to maintain a running total of all digits presented. As each digit was presented the participant was to speak the new total into a microphone located below the line of sight between the participant and the monitor. The beginning of the verbal response triggered a voice-onset key attached to the computer. Three seconds after the onset of a verbal response (i.e. approximately two seconds after its completion) the computer presented the next number in the sequence. The task continued for a total of twenty digit presentations. Digits were selected at random, subject to the constraint that the correct total for each sequence never be more than 222 (i.e. three digits in length).

The Auditory Probe Task

Probe tones were presented during five randomly selected trials in each sequence. The probe was a 130 HZ tone with a duration of 60 msec., presented 100 msec. after onset of the visual stimulus. The participant responded to the tone by pressing a button. Participants

COPY AVAILABLE

were instructed to press the button before making a verbal response.

The participants were instructed to say 'stop' when they became aware of making an error. After stopping, they were asked to explain what the error was (e.g. forgetting to press the button, making an addition fact error). They then made a corrected response. Following the corrected response the remaining digits in the sequence were presented.

General description of behavior

Response latencies for experimental trials were generally between 300 and 2500 msec. and overall error rates were below 2 percent.

Perfect performances on the pretest and low error rates in experimental sessions demonstrated the subjects knew how to do the tasks. However, a number of different types of errors were observed. The following types of errors were considered:

1. Arithmetic fact errors. Even though the participants thoroughly learned the rules of addition for trinary arithmetic, they occasionally made errors,

just as most adults do with decimal arithmetic.

2. Speaking the stimulus digit rather than adding it to the running total. An example would be responding '2' to the second digit in the sequence 1, 2. Participants reported seeing the stimulus and speaking it without considering addition or the running total.

3. Not pushing the button in response to the probe. Frequently this error was not detected by the participants.

4. Pushing the button after making a verbal response. Participants reported that it was not a 'conscious' effort to correct for not pushing the button before the verbal response.

5. Forgetting the stimulus digit or the running total. Only one participant made this type of error and only when he interrupted himself to correct a previous error.

6. Carry errors, i.e. a failure to carry the 1 to the next column to the left and to do the appropriate addition. For completeness, we note that these errors are not the only errors that could be made. Hitch (1978) has reported more complex carry errors that cannot be distinguished from primary arithmetic fact errors.

7. Transposition errors. Transposition errors (e.g. saying 221 when the correct response is 212)

BEST COPY AVAILABLE

are logically possible but we did not observe any.

The participants would often detect their own errors as they made them. This will be called "error trapping."

To summarize, the participants generally, but not always, managed to remember the things they needed to remember. The fact that they often became aware of an error after making it indicates that there was a good deal of self monitoring of behavior. The problems solver's internal control of information seems to be an essential part of real-time problem solving. These processes were reflected in the simulation, which will now be described.

The Simulation

Terminology

As noted, the model consists of a set of productions that are executed by the functional architecture described by Hunt and Lansman (1983). The modified architecture will be described first. We then describe the productions that it executed.

BEST

Functional Architecture The term 'functional architecture' refers, collectively, to mechanisms required for production selection and execution. The term will be used to include a specification of the major structural components of the system (e.g. the existence of input buffers and a working memory) and constraints on the nature and size of those components; and the nature of the pattern matching mechanism.(1)

The architecture consists of five structural components: an auditory input buffer , a visual input buffer , a motor input buffer , a working memory (WM) and a long term memory (LTM). The input buffers correspond to external sensory channels. They and the WM constitute a 'blackboard' area that contains the patterns that form an internal representation of the current situation. The WM is divided into five areas (or codes), semantic, visual, motor, auditory and a 'metacognitive' area. The LTM contains the productions that respond to patterns in each of the blackboard areas. Figure 1. shows the relation between components, and indicates permissible flows of information between them.

COPY AVAILABLE

Fig. 1 here

The selection of the productions to be executed is driven by the data in the blackboard area. The pattern segment of each production is continuously compared with the information in the input buffers and in WM. The extent of agreement between a production's pattern and the data on the blackboard determine the level of activation of the production. Productions are also activated by spreading activation from related productions. Thus the productions can be thought of as being linked together in a semantic network of associated concepts. Negative associations are used when productions are logical alternatives to each other (See Hunt and Lansman for a discussion of the algorithms for pattern matching and the rules for linking productions into a network). When a production's activation level exceeds a threshold and is sufficiently greater than the activation level of other productions the action part of the production is executed.

The architecture used in this study modifies Hunt and Lansman's system in three ways.

(a) In Hunt and Lansman's model information stayed in a blackboard area unless it was specifically replaced. In our model the 'clarity' of information in WM decays over time unless it is explicitly refreshed. To represent the decay, a decay rate parameter, d(0>d>1) was associated with WM. Instead of assuming that an object would be matched (to some degree) to a pattern regardless of its time in WM the match was computed with probability Pr(e|t) for an object that had been in WM for t cycles without being refreshed. Pr (e|t) was defined by the equation $Pr(e|t) = (1-d)^t$, t=1,2,3 ... where t is the number of discrete time cycles since the object was placed in WM at time t=0.

(b) An additional motor code was added, with its own buffer area and channel in WM.

(c) A metacognitive area was added to WM. The productions responding to patterns appearing in this area dealt with qualitatively different processes than those responding to patterns in the other areas of WM. The patterns in other areas represented actions needed to solve the addition or the probe response task. The patterns in the metacognitive area dealt with the system's view of its own problem solving processes. As can be

EST COPY AVAILABLE

seen, this simulation contains two separate models for problem solving. One is a task model that is specific to the problem at hand; here either arithmetic or probe responding. The second is a model for monitoring problem solving actions.

We believe that this is characteristic of real time problem solving. The problem solver must be able to monitor where he (it) is in the problem solving process at any time.

The task model contains the following submodels:

1.An addition task submodel containing productions required for trinary addition.

2.An auditory probe task submodel containing the productions that recognize the probe tone.

3.An input/output submodel containing those productions that translate patterns from the sensory buffers into areas of WM, and those productions that translate certain patterns appearing in WM into external responses. These patterns will have been produced by the addition or auditory probe submodels or the metacognitive

error trapping submodel.

The metacogn.tive model contains:

1. An articulatory submodel containing productions that detect a need to rehearse information currently in WM. Rehearsal is accomplished by moving patterns into auditory WM, recognizing them, and then placing the product of pattern recognition back into a semantic WM area. These productions are used to simulate subvocal articulation.

2. A goal insertion submodel containing productions that insert goal patterns into the semantic area of WM. The goals are used to guide the task model.

3. An error trapping submodel containing productions that identify specific errors, such as not pressing the button. These productions then produce patterns indicating the type of errors that are then output by the verbal output system.

Productions.

Although it is technically correct to think of our

T COPY AVAILABLE

simulation program as being a production system, the usual if-then notation of production systems does not capture adequately the important interactions between productions. To do this we use an abbreviated form of the dataflow graph notation presented by Sowa (1984).

Dataflow graphs.

A dataflow graph is a finite, connected, directed, bipartite graph with the following characteristics:

a. The nodes are partitioned into concept and actor nodes.

b. Arcs may connect concepts and actors, but arcs between concepts and concepts or actors and actors are not permitted.

c. An arc directed from a concept c to an actor a is termed an input arc of a and the concept c an input concept of a.

d. An arc directed from an actor a to a concept c is termed an output arc of a and the concept c an output concept of a.

e. An actor must have one or more input concepts and must have exactly one output concept.

f. A concept c may be an input concept for more than one actor.

To map from patterns and actions to concepts and actors, let patterns be represented by input concepts and actions by actors and output concepts.

The data flow graph illustrates how the productions relate to each other. This is shown in Figure 2, which illustrates some key features of the graphs. Concepts are drawn as rectangles, actors as diamonds, and the arrows on arcs indicate direction. This graph has two actors, nodes a1 and a2. These correspond to the action parts of two productions, p1 and p2. The concept nodes c1 and c2 correspond to the pattern part of production p1, and the concept nodes c2 and c3 correspond to the pattern part of production p2. An interpretation of this graph in terms of productions is that the action of production p1 is executed in response to the pattern consisting of concepts c1 and c2. Concept c3, resulting from the

BEST COPY AVAILABLE

action of production p1 , together with concept p2 constitute the pattern of production p2 , and concept p4 results from the action of production p2 being executed.

Figure 2 here

Concepts : Concepts are concatenations of primitive features (symbols) that are either objects or labels.

a. Objects . Objects are internal symbols that correspond to physical stimuli. The objects we deal with include:

1. The trinary digits.

2. The probe tone.

3. Columns These are the units 3s, 9s, and 27s digit positions of a trinary integer.

4. The (perception of the) fingerpress.

5. The null object, i.e. the absence of an object.

BEST

Objects may be concatenated to form compound objects. For example, the objects 2 and 1 concatenate to form the object 21 . The only objects that will be compounded are digits and columns. The occurence of an object symbol (hereafter referred to as an object) in a blackboard area can be interpreted as the occurence of a specific member of a set. For example, the digit 2 is a member of the set of integers . (as well as a member of the set of numbers greater than zero etc.)

Labels . Labels are primitive features that are attached to objects by the actions of specific actors, thus assigning additional meaning to the objects to which they are attached. In general, labels specify the cognitive roles that the objects play in the task at hand. For example, when the object 2 is labeled as an integer it is identified as belonging to the set of integers, thus enabling further processing specific to integers.

The labels used in the model are:

1. Stimulus - classifies an object appearing in an input buffer a stimulus.

2. Integer - classifies a physical digit as a number.

COPY AVAILABLE

3. Addend - classifies an integer digit as an addend. In order for an integer to serve as input for an addition process it must be classified as an addend.

4. Total - classifies the integer output from an addition process as a sum.

5. Carry - classifies 1 as the carry digit in an addition.

6. Ur .a - classifies a column or a single digit integer as being in the rightmost position in a multi-digit integer.

7. 3s - (see Units).

8. 9s - (see Units).

9. 27s - (see Units).(2)

10. Zero - classifies an integer as zero. This label activates productions that deal with the number zero.

11. Blank - classifies a column as blank (i.e. there is no integer labeled as belonging to that column).

12. Not - may be combined with other labels to classify an object as not belonging to a particular set (e.g. Not Zero or Not Blank).

13. Input - combined with a label that designates an input buffer (e.g. Auditory) to classify an object as input to the system.

14. Output - combined with a label that designates an output system (e.g. Verbal) to designate an object for output (i.e. to be translated into an external response).

15. Auditory - combined with the label Input, classifies an object as having appeared in the auditory buffer (i.e. as Auditory Input).

16. Visual - (see Auditory).

17. Somatic - (see Auditory).

18. Verbal - combined with the label Output, classifies an object as verbal output (i.e. enables actors that will translate the object into an external verbal response).

19. Manual - (see Verbal).

20. Same - classifies two objects as belonging to

the same object type (e.g. if two objects are both labeled as the Units Integer then they might be labeled as Same).

21. Different - (see Same).

22. Old - when a total for the addition task has been output as an external verbal response that total is labeled Old to distinguish it from the New total that will be computed from the old total and a stimulus integer (see New).

23. New - when a new total has been computed for the addition task it is labeled as New. (see Old).

24. Null Label - absence of a label.

Actions The actions permitted to productions are:

a. Label - Attach one or more labels to one or more objects. For example, a particular actor labels the object 2 as an Integer. Labeling produces a refined definition of an object that may cue subsequent actors.

b. Insert - Place an object in an area of WM. For example, one of the 'arithmetic fact' actors recognizes

the pair of integer addends 1 and 1, and places object 2 , labeled as a sum, in the semantic area of WM.

c. Translate - Move an object from an input buffer to an area of WM or from one area of WM to another. A translation from an input buffer to an area of WM retains the original sensory code of the object. Certain actors also translate objects from WM to external verbal and manual output systems. (These output systems are not further defined in our model).

Individual actors may act upon more than one object and may execute more than one of the three basic actions on the object(s). For example, the arithmetic fact actor that Inserts object 2 into the semantic area of WM also Labels that object as New and as Total (4).

Patterns, concepts, and the contents of the input buffers and WM.

Each blackboard area contains discrete primitive features, consisting of objects and their attached labels. An object and its labels must always be in the same blackboard area.

A concept consists of a concatenation of labels and

BEST COPY AVAILABLE

objects. A concept may contain one or more labels, with or without the object to which the labels are attached, or may contain an object, with or without the labels attached to it.

A pattern consists of one or more concepts. The concepts that make up a particular pattern must be in a single blackboard area.

For example, consider a situation in which the current total is 1 and the digit 2 is shown. Once the stimulus digit was recognized WM would contain the following features:

a. stimulus & integer & 2 & not zero.

b. old & total & units & integer & 1 & not blank

These are instances of the concepts: (stimulus & integer & not zero) and (old & total & units and not blank). Each concept is defined in terms of the labels attached to a single object in this example (but not generally) and neither concept includes the object. The two concepts are the input concepts (on the pattern) for an actor that labels each object as an addend. After this actor executes, the contents of the semantic area of WM would be:

a. stimulus & integer & 2 & not zero & addend

b. old & total & units & integer & 1 & not blank & addend.

The augmented description would trigger the arithmetic fact rule for adding 2 and 1.

Behavior of the Simulation

This section presents a dynamic description of the simulation. First, the addition and auditory probe task submodels will be presented, using block diagrams. Then, a detailed walkthrough of a specific example will be given, with the aid of a dataflow graph. Finally, we describe the actions of the various "metacognitive" submodels; articulation, goal insertion and error trapping.

The Addition Submodel

Figure 3 provides an overview of the addition task submodel.

Block 1 . " Evaluate Input ": Input is 'sensed' by the appearance of a (digit) object in the visual buffer. The symbol is transferred into the visual area of WM. The

BEST COPY AVAILABLE

digit is labeled as an integer, and as a stimulus. It is then translated into the semantic area of WM, where it is labeled as zero or not zero. Each transfer or labeling is achieved by production selection and execution.

This is the point at which the first of several possible errors may occur. The productions that can cause it to 'speak' the name of a digit may be activated instead of those that deal with the digit as an addend. Looked at another way, enroute to creating the addend pattern the system comes close to creating patterns that trigger the speaking actions of block 3 of Figure 3.

Block 2, Labeling : The old total is labeled as blank or not blank. Based on this information and whether the stimulus is zero or not a decision is made either: to label the stimulus as verbal output (Block 3), to label the old total as verbal output (Block 3) or to add the timulus and the old total to form a new total (Block 5). The last decision is the most interesting case, so it will be considered first.

Block 5, Adding : The stimulus and the unit column digits are labeled as addends. The sum of the two digits

is inserted in the semantic area of WM and labeled as the new units integer. If a carry is necessary, and the next column (3s column) is labeled as blank or not blank, the integer 1 is labeled as the 3s integer and the 3s column is then labeled as not blank. If the 3s column is already labeled as not blank, then the 3s column integer and the carry integer (i.e. 1) are labeled as addends. The sum of these two digits is inserted in the semantic area of WM and labeled as the new 3s integer. The labeling and insertion process continues across columns until the addition process is completed and a new total is formed. (Note that columns to the left of the units column are not processed unless there is a carry.)

Block 6.. Output The new total is labeled for output as a verbal response.

The output label activates the productions. Control is in Step 7, where a verbal response is created. Note that Step 7 may be activated by any of several prior steps.

Fig. 3 here

BEST COPY AVAILABLE

The Auditory Probe Submodel

Figure 4 is a block diagram that represents the auditory probe task submodel.

Block 1 Recognition. When a tone appears in the auditory buffer it is translated into the auditory area of WM. There it is labeled as the stimulus tone and translated into the semantic area of WM.

Block 2. When the tone is recognized in semantic WM the fingerpress object is inserted in semantic WM as a goal and labeled as a manual response output.

Block 3 Response Production. When a stimulus in semantic WM is labeled as an output the external manual response is made.

Fig. 4 here

An Illustrative Example .

Figure 5 displays parts of a dataflow graph summarizing the action of the addition task submodel for a single trial. Suppose that the old total is 12 (Decimal 5) and the digit 1 appears. The following actions will take place. The number or letter in parentheses before each comment locates the action in the graph of Figure 5.

(1) The object 1, labeled as a stimulus integer is translated from the visual to the semantic area of WM.

(2) Actor 2 labels the stimulus integer 1 as not zero.

(3) The old total integer (for this example 12) is not blank.

(4) Because the stimulus integer is not zero, and the old total is not blank Actor 4 labels the stimulus integer and the units integer of the old total as addends. The trinary addition '1 + 2' is to be performed.

(5) Actor (5), an arithmetic fact, carries out the addition. Note that actor 5 has objects labeled as addends as its input concept. The object 0 is placed in WM labeled as an integer total, and the object 1 is placed in WM, labeled as a carry integer.

(6) The old 3s column is not blank, as the old total is 12.

BEST COPY AVAILABLE

(7) Because the stimulus integer and units integer have been labeled as addends, and there is a carry integer and the 3s column is not blank, actor (7) labels the carry integer (i.e. 1) and the 3s integer (i.e.1) as addends.

(8) Actor (8), an arithmetic fact, carries out the addition and inserts object 2 in semantic WM as an integer and new column total (in this case the 3s column). Note that there is no carry integer.

(9) Because the 9s column and 27s column are blank, and there is no carry integer, actor (9) labels the column totals 2 (i.e. 3s) and 0 (i.e. units) as the new total, and labels them as verbal output.

Fig. 5 here

The Metacognitive Behavior .

The metacognitive model is a set of procedures that monitor the performance of the other models. Three types of data are monitored. These are the quality of data in working memory, the progress of each task model toward its problem solving goals, and the existence of conditions in WM indicating that an error has occurred. The model itself is divided into three submodels, each of which

accomplishes one of these functions.

The articulatory submodel : This submodel contains processes that protect the information in WM against decay. The submodel contains actors that translate objects from semantic area to the auditory area of WM and back again. The action of translating has the effect of refreshing the object, by inserting a fresh copy into WM. The actors that translate from semantic to auditory are triggered by an object's being in the semantic area for some time without being acted upon (i.e. Labeled or Translated).

The goal insertion submodel : Actors in this submodel insert goals into the semantic area of WM. The goals are concepts consisting of a null object and various labels that, collectively, represent a desired future state. The goal would be attained if the labels could be attached to an object. The goal insertion actors execute after an object(s) has been in the semantic area for some time and usually has been refreshed (translated) by the articulatory actors one or more times without otherwise being acted upon. (This condition arises if processing on a task is stalled). For example, suppose that the stimulus integer and the units integer are both labeled as

BEST COPY AVAILABLE

addends and have been so labeled some time. After these objects have been translated by the articulatory submodel actors one or more times a goal insertion actor may insert a null object labeled as a 'new column total' into the semantic area.

The execution of the goal insertion actors raises the activation level of the relevant task submodel actors as a result of their linkages in the association network. This guides processing toward satisfaction of the appropriate goal.

Error trapping submodel . Various actors in this Submodel identify errors (i.e. patterns consisting of incorrect sets of labeled objects), label the objects as errors, and make a verbal response that identifies the error. For example, if a tone, labeled as the auditory probe, is in the semantic area of WM, and an integer, labeled as a new total and as auditory input, (i.e. the new total has been spoken and is heard) is in WM, but there is no fingerpress, labeled as motor input, (i.e. the key pressed and the motor response felt) in the semantic area, then the simulation "knows", in effect, that a verbal response was made before responding to the auditory probe, which is an error. This condition is the input pattern for an

actor that identifies the error.

Comparison to Human Data

To evaluate the success of the model in accounting for the participants' performance we compared both qualitative and quantitative aspects of human behavior to the simulation program's behavior. The simulation can clearly perform the addition and probe response tasks.

Depending on the value of the internal parameters the simulation can produce all seven of the error types described earlier. We have not explored settings that produce carry and transposition errors, as these do not appear in our data. Table 1 lists the error types, the observed frequencies for our participants, whether or not the error type was trapped by participants, and the mechanisms in the model that produce each error type. The model is also able to trap all the error types trapped by the participants.

Table 1 here

Comparing reaction time data provides a more

BEST COPY AVAILABLE

quantitative contrast between the model and the human data. Two types of responses can be made, additions and key press responses to the probe and signal. Table 2 lists each participant's median reaction times to addition problems under four conditions: when a probe was present and not present, and when a carry operation was and was not required. The table also lists the median reaction times for responses to the probe tone when the concurrent addition trial did or did not require a carry.

The table also shows the number of discrete time cycels the simulation required to make the same responses. The simulation's responses also have to be averaged because there is some variation due to internal noise in the system.

Table 2 here

In order to facilitate a comparison between the simulated and actual data all data were converted to times (or cycles) relative to the time required to respond to a probe while doing an addition without a carry. This was the quickest response for both the simulation and the human subjects. The rescaled numbers are shown in Table

3. The simulation generally did a good job of predicting the relative reaction times for each of the six conditions.

Table 3 here

EST COPY AVAILABLE

Comments

A. No error producing productions were used. This is a contrast to the work of Brown and Burton (e.g. Brown and Burton, 1977), who explicitly used 'buggy' productions to model children's arithmetic errors. We assumed that our adult subjects knew how to do the experimental tasks. Thus we are forced to conclude that errors arise from the misapplication of correctly formed productions. The errors that our model permits arise from the mechanisms listed in Table 1. As we have noted, these are sufficient to produce all errors that were observed.

B. The complexity of the productions that we used was severely constrained.

1. Decisions that direct the behavior of the system were made by the interaction of productions rather than by the action of individual productions. For example, the decision to make an arithmetic response by naming the stimulus integer, the old total or with a new total was made by the actions of the several productions that determine if the stimulus was zero, if the old total

was blank, and so forth.

2. The patterns that initiate productions were relatively simple, compared to those used in other simulations (e.g. Anderson, 1983). The most complex patterns that occur in our model are those used by the productions that trap errors. These do not require more than four concepts in a single pattern.

3. Patterns were constructed from a small set of primitive features. This is widely accepted as a requirement for production system models (Newell, 1980). The sharing of features across concepts and patterns and the possibility of confusion arising from misidentification of them is a basic mechanism in our model for producing errors.

4. The actions taken by productions were simple and limited in number.

5. There was no branching within individual productions. At least two versions of this constraint are possible. The weak version constrains a production to take a similar action on any set of target objects (e.g. doubling single digit integers). The resulting output then varies with the input. Such a production, instantiating a nontrivial function, must not only match a pattern but also distinguish between inputs. The strong version of the no branching constraint requires individual

BEST COPY AVAILABLE

productions to produce invariant output.

We adopted the stronger constraint. This leads to a significant difference between our model and John Anderson's ACT* model (1983), which utilizes local variables and thus follows the weak constraint. In this respect our model is like that of McClelland and Rumelhart (1981).

C. Patterns and actions were all object-oriented. This constraint limits the denotational power of productions by requiring a physical object reference for abstract concepts (e.g. the concept that an integer is not zero).

D. Goals were structured and processed in a fashion similar to other concepts. In our model goals are concepts constructed from the same primitive features as other concepts and processed in the same way. As a result, the model is entirely data-driven at the level of individual productions.

Significance. In the course of our work we found several ways in which the architecture of the original Hunt and Lansman model was insufficient for the job of modeling the

complex real-time problem solving studied here. We have modified the architecture in the ways described earlier.

As in any simulation, some aspects of the present model are obviously tied to the particular arithmetic and probe tasks we have studied. These specializations are in the production systems, and are of little general importance. The effects of the restrictions on production system design are more interesting. We have shown that a relatively complex real-time task can be modeled by a production system mechanism that conforms to a substantial number of constraints. The major importance of this work is that it provides a further link between the 'time-less' problem solving studied in most simulations of cognition and the highly simplified, but tightly time bound, situations used to develop models of human information processing.

BEST COPY AVAILABLE

Footnotes

1. Our use of the term functional architecture is similar to Anderson's usage (1983) and somewhat more limited than Pylyshyn's (1984). Pylyshyn's usage includes general characteristics of productions such as constraints on the kinds of actions that productions can execute and what counts as a semantic primitive (i.e. pattern feature). Although we would concur with Pylyshyn that such things are properly 'architectural' our exposition of the model is simplified if we present them in our discussion of Productions (section III B) and in our general discussion (section VI A) rather than as part of the functional architecture.

2. The longest correct total for the experimental task was three digits. Occasionally subjects made errors that led to four digit responses (i.e. an integer in the 27s column).

3. This seems to permit very powerful actors in the model, but is necessary for keeping the simulation within practical computing limits. Each actor that executes more

than one basic action or acts on more than one object could be replaced by separate actors each executing only one basic action and acting on a single object, without affecting the logic of the simulation.

8. We would like to compare the relative frequency of each error type for the simulation with the frequencies produced by participants, however, the partipants' overall error rate was less than two percent and such a comparison would require several thousand trials for the simulation. This would be prohibitively expensive in computing costs. For example, to run a total of 3200 trials as our four subjects did would require over 80 hours of system operating time on the VAX/780 that we used for the simulation.

BEST COPY AVAILABLE

REFERENCES

Anderson, J.R. (1983) The Architecture of cognition. Hillsboro,NJ: Erlbaum Associates.

Brown, I. and Poulton, E.C. (1961) "Measuring the spare 'mental capacity' of car drivers by a subsidiary task." Ergonomics. 4. 35-40.

Brown, J.S. and Burton, J.R.(1978) "Diagnostic models for procedural bugs in basic mathematical skills." Cognitive Science. 2. 155-192.

Hitch, G. "The role of short term working memory in mental arithmetic." (1978) Cognitive Psychology. 10. 302-323.

Hunt, E. and Lansman, M. (1984) "Performance in Dual Tasks." Technical Report No. 84-2. Sponsored by Personnel and Training Research Programs, Psychological Sciences Division, Office of Naval Research.

McClelland, J.L. and Rumelhart, D.E. (1981) "An interactive model of context effects in letter perception: Pt. I. An account of basic findings." Psychological Review. 88. 375-407.

Pylyshyn, Z.W. (1984) Computation and Cognition. Cambridge, MA: Bradford Books, MIT Press.

Reitman, J.S.(1974) Without surreptitious rehearsal, information in short term memory decays. Journal of Verbal Learning and Verbal Behavior. 13. 365-377.

Schneider, W., Dumais, S. and Shiffrin, R.M. (1984) "Automatic and control processing and attention." In Parasuraman, R. and Davies, D.R. (Ed.) Varieties of Attention. Orlando: Academic Press, Inc.

Schneider, W. and Shiffrin, R.M. (1977) "Controlled and automatic human information processing: I. Detection, search and attention." Psychological Review. 84. 1 - 66.

Sowa, J.S. (1984) Conceptual structures: Information processing in mind and machine. Reading, MA: Addison-Wesley Company.

BEST COPY AVAILABLE

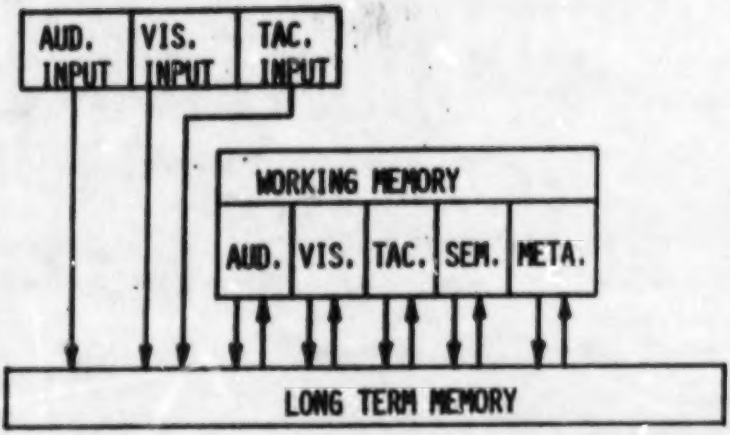
AUD. INPUT
VIS. INPUT
TAC. INPUT
WORKING MEMORY
AUD.
VIS.
TAC.
SEM.
META.
LONG TERM MEMORY

Figure 1

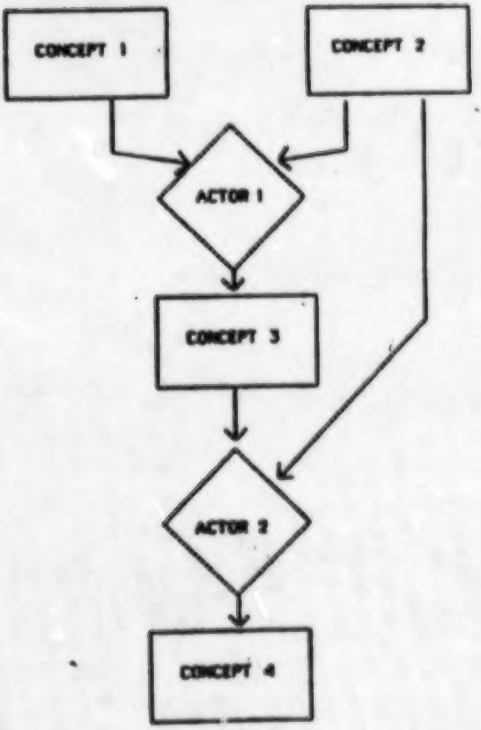

Figure 2

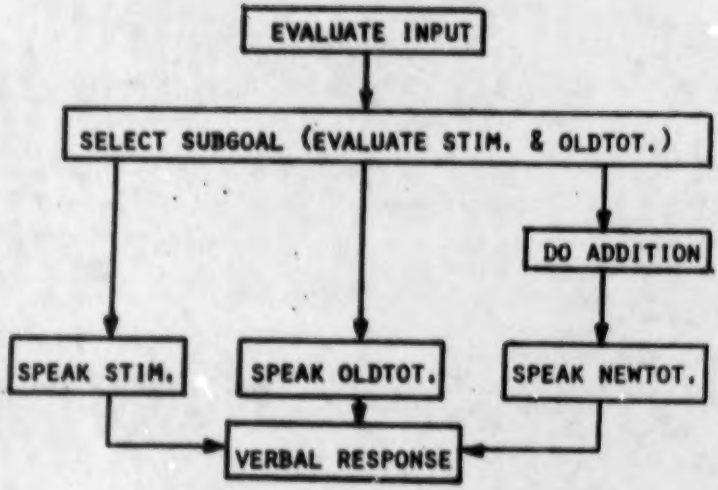
EVALUATE INPUT
SELECT SUBGOAL (EVALUATE STIM. & OLDTOT.)
DO ADDITION
SPEAK STIM.
SPEAK OLDTOT.
SPEAK NEWTOT.
VERBAL RESPONSE

Figure 3

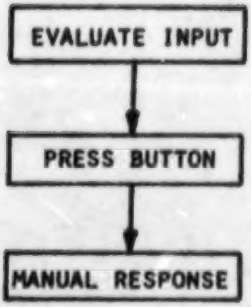

Figure 4

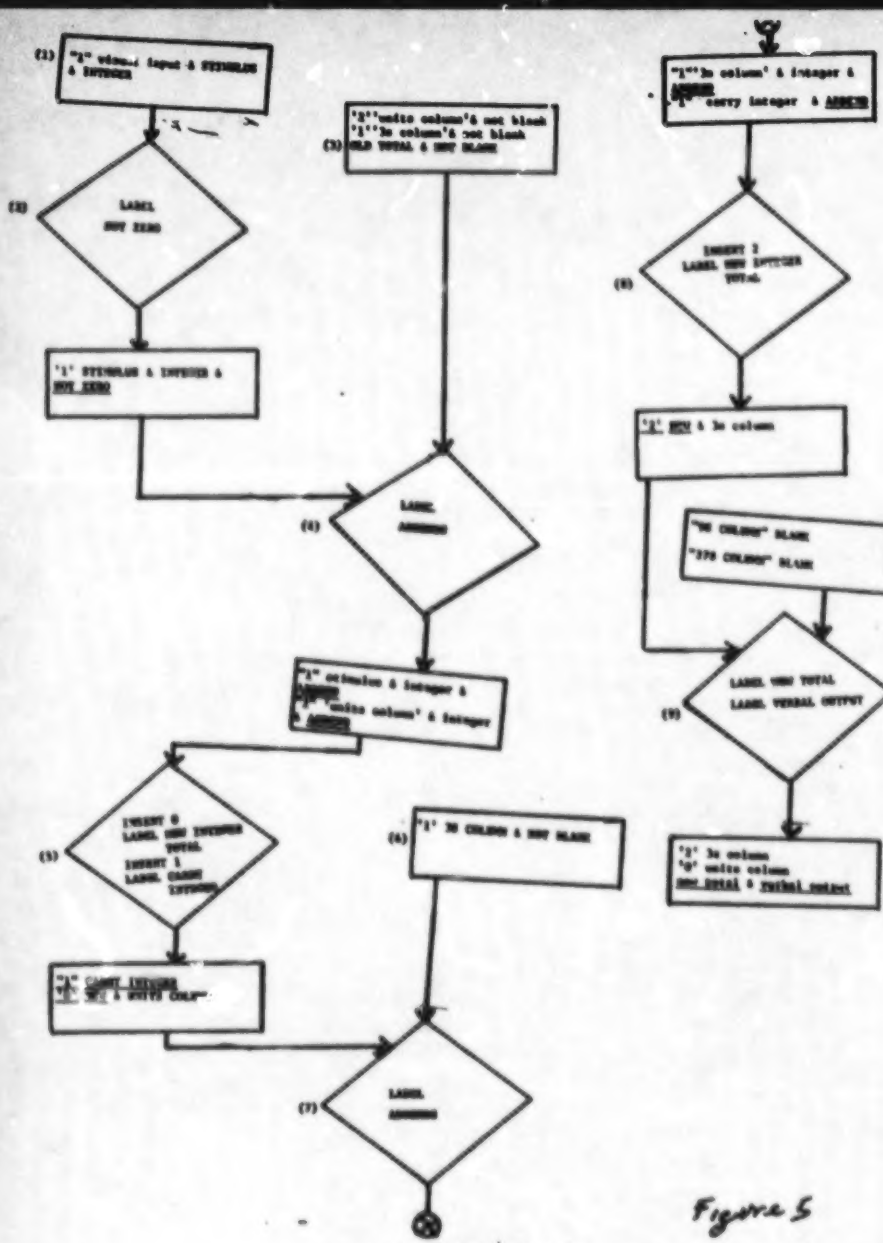

Figure 5

Notes for Figure 5

Labels in input concepts are all uppercase

Labels in output concepts are underlined

Labels otherwise are lower case and included to improve readability

Objects in input concepts are in double quotes

Objects in output concepts are underlined

Objects otherwise are in single quotes and included to improve readability

BEST COPY AVAILABLE

Figure Captions

Figure 1. Structural components of the functional architecture and permissible flows between them.

Figure 2. An abstract example of a dataflow graph.

Figure 3. Block diagram of the addition task submodel.

Figure 4. Block diagram of the auditory probe task submodel.

Figure 5. A dataflow graph for a specific example of the addition task submodel processing a single trial.

Table 1

Participant's Errors and Mechanisms in the Model for Simulating Them

Error Type	Occurance In Participant's Data	Trapped By Participant's	Mechanism In Model
1. Speaking the Stimulus	24 (.41)	Yes	Priming of Production From Previous Trial.
2. No Probe Response	13 (.22)	Yes	Decay of Probe, Competition Between Tasks.
3. Probe Response After Addition Response	9 (.16)	Yes	Competition Between Tasks.
4. Addition Fact Errors	9 (.16)	No	Confusability of Addends.
5. Forgetting Stimulus or Old Total	4 (.07)	Yes	Decay of Integers.
6. Carry Errors	0	NA	Persistence of Pattern in WM.
7. Transposition Errors	0	NA	Confusability of Integers.

Note. Proportion of total errors in parentheses.

Table 2

Median Reaction Times for Participants (in ms) and the Simulation (in time cycles)

Condition	Subject				Simulation
	1	2	3	4	
Probe -	451	662	594	712	6
No Carry	(137)	(138)	(133)	(136)	(16)
Probe -	462	768	620	765	7
With Carry	(60)	(62)	(57)	(60)	(9)
Addition - No	586	695	809	964	9
Carry, No Probe	(404)	(412)	(396)	(397)	(48)
Additon - No	863	1156	1214	1344	11
Carry, With Probe	(137)	(138)	(133)	(136)	(16)
Addition - With	1578	1436	2027	2668	17
Carry, No Probe	(195)	(188)	(198)	(198)	(27)
Addition - With	2209	1660	2483	2814	22
Carry, With Probe	(60)	(62)	(57)	(60)	(9)

Note. Number of observations in parentheses.

Table 3

Scaled Reaction Times for Participants and the Simulation

Condition	Subject				Simulation
	1	2	3	4	
Probe - No Carry	1.0	1.0	1.0	1.0	1.0
Probe - With Carry	1.0	1.2	1.0	1.1	1.2
Addition - No Carry, No Probe	1.3	1.0	1.4	1.4	1.5
Addition - No Carry, With Probe	1.9	1.7	2.0	1.9	1.8
Addition - With Carry, No Probe	3.5	2.2	3.4	3.7	2.8
Addition - With Carry, With Probe	4.9	2.5	4.2	4.0	3.7

Note. Values for each participant and the simulation are scaled separately in terms of the median Probe- No Carry reaction times, which were in each case the shortest median times.